# BEST AVAILABLE IMAGES

Defective images within this document are accurate representation of The original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

## IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.**

E5934

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-334119
(43)Date of publication of application : 17.12.1993

(51)Int.Cl. G06F 11/28
G06F 3/14

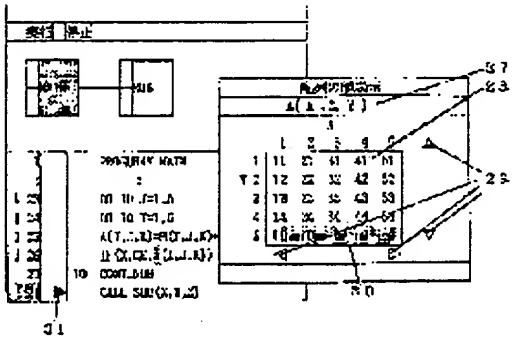
(21)Application number : 04-144139
(22)Date of filing : 04.06.1992

(71)Applicant : HITACHI LTD
HITACHI TOHOKU SOFTWARE KK

(72)Inventor : KAWAGUCHI SOTARO
KIKUCHI SUMIO
SATO MAKOTO
YAMADA SHIGEMI

## (54) METHOD FOR DEBUGGING PROGRAM

(57)Abstract:
PURPOSE: To improve debugging efficiency by two-dimensionally displaying the values of array elements at the time of interrupting a program.
CONSTITUTION: An interruption point 31 set in a module execution sentence and a user instructs the reference of elements of an array A. When a subscript setting part 27 sets up the x and y axes of a dimension to be variable, a program debugger extracts the leading address of the array A included in analytical information, finds out the address of respective elements in the array A and reads out and two-dimensionally displays the element value of using the addresses as keys while referring to a load module. At the time of displaying the element values of the array A, invalid data (value '0') are also displayed on a display part 30.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（Ａ）　(11)特許出願公開番号

特開平5－334119

(43)公開日　平成５年(1993)12月17日

| (51)Int.Cl.[5] | 識別記号 | 庁内整理番号 | ＦＩ | 技術表示箇所 |
|---|---|---|---|---|
| Ｇ０６Ｆ　11/28 | | Ｐ　9290－5Ｂ | | |
| 3/14 | ３１０ | Ｅ　7165－5Ｂ | | |

審査請求　未請求　請求項の数１(全 11 頁)

(21)出願番号　特願平4－144139

(22)出願日　平成４年(1992)６月４日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(71)出願人　000233538
日立東北ソフトウェア株式会社
宮城県仙台市青葉区一番町２丁目４番１号

(72)発明者　川口　荘太郎
宮城県仙台市青葉区一番町二丁目４番１号
日立東北ソフトウェア株式会社内

(72)発明者　菊地　純男
東京都国分寺市東恋ケ窪１丁目280番地
株式会社日立製作所中央研究所内

(74)代理人　弁理士　鈴木　誠

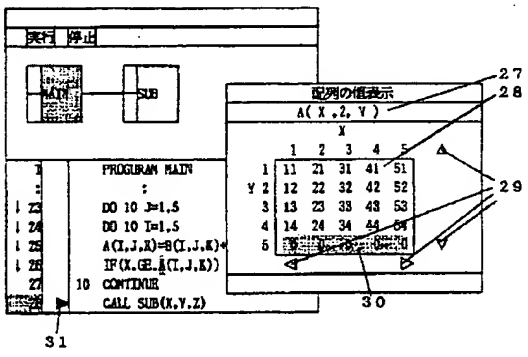
最終頁に続く

(54)【発明の名称】　プログラムデバッグ方法

(57)【要約】

【目的】　プログラム中断時における配列要素の値を２次元表示することによりデバッグ効率を向上させる。

【構成】　モジュールの実行文に中断点３１を設定し、利用者が配列Ａの要素の参照を指示する。添字設定部２７で可変とする次元のｘ軸、ｙ軸を設定すると、プログラムデバッガは、解析情報中の配列Ａの先頭アドレスを取り出し、配列Ａの各要素のアドレスを求め、そのアドレスをキーにして、ロードモジュールを参照して、要素値を読み出して二次元表示する。配列Ａの要素値の表示において、表示部３０には不正なデータ（値０）も表示される。



1

【特許請求の範囲】

【請求項１】　プログラムの実行を中断させつつデバッグ作業を行うプログラムデバッグ方法において、配列要素中の表示すべき二次元の添字を表示画面上の所定領域に設定し、該設定された添字を逐次変化させたときの対応する配列要素の値をロードモジュールから読み出して二次元表示し、該表示された要素の値を一部設定変更してプログラムを実行することを特徴とするプログラムデバッグ方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、対話環境で行うプログラムのデバッグ方法に関し、特にプログラム実行中断時における配列要素を二次元表示するデバッグ方法に関する。

【０００２】

【従来の技術】従来、プログラム実行時における配列の値を参照する場合、ダンプリストを用いて配列要素に対する記憶場所の計算を行うことによって、配列の値を得ていた。このため、利用者の作業負担が大きくなり、デバッグ効率が著しく悪かった。そこで、プログラムの実行を中断し、配列の値の参照が可能なプログラムデバッガが種々開発され、実用に供されている。

【０００３】

【発明が解決しようとする課題】しかしながら、上記した従来のデバッグ方法では、プログラム実行の中断時に参照したい配列の名称と要素を個々に指定しなければならず、特に、不連続なアドレスにある複数の配列要素の値を一度に表示することができないため、プログラムのデバッグ作業を効率的に行うことができないという問題があった。

【０００４】本発明の目的は、プログラム中断時における配列要素の値を２次元表示することによりデバッグ効率を向上させたデバッグ方法を提供することにある。

【０００５】

【課題を解決するための手段】前記目的を達成するために、本発明では、プログラムの実行を中断させつつデバッグ作業を行うプログラムデバッグ方法において、配列要素中の表示すべき二次元の添字を表示画面上の所定領域に設定し、該設定された添字を逐次変化させたときの対応する配列要素の値をロードモジュールから読み出して二次元表示し、該表示された要素の値を一部設定変更してプログラムを実行することを特徴としている。

【０００６】

【作用】利用者は、モジュールの実行文に中断点を設定し、配列の要素の参照を指示する。プログラムデバッガの操作画面上の添字設定部で可変とする次元のｘ軸、ｙ軸を設定すると、プログラムデバッガは、解析情報中の配列の先頭アドレスを取り出し、その配列の各要素のアドレスを計算する。計算されたそのアドレスをキーにして、ロードモジュールを参照して、要素値を読み出し二次元表示する。これにより、配列の各要素を任意に参照することが可能となり、デバッグ作業を効率的に行うことができる。

2

【０００７】

【実施例】以下、本発明の一実施例を図面を用いて具体的に説明する。図２は、本発明のデバッグ方法が適用されるデバッガのシステム構成図である。コンパイラ２は、ソースプログラムが登録されたソースファイル１を入力して、字句解析・構文解析を行い、ソースプログラムの解析情報３を生成すると共に、コード生成処理によってオブジェクトプログラム４を出力する。さらにオブジェクトプログラム４をリンケージエディタ５によって連系編集し、ロードモジュール６を生成する。

【０００８】利用者がプログラムデバッガ７を起動すると、プログラムデバッガ７は、解析情報３とロードモジュール６、及びソースファイル１を入力して、コンソール８に、モジュールの引用関係を示すコールグラフとモジュールのソースプログラムを同時に表示する（初期表示）。

【０００９】また、後述するように、この初期表示から、利用者はコールグラフのレイアウト表示、拡大表示、及びプログラム構造の輪郭図を表示するマッピング表示などを用いながら、任意にモジュールの入口、出口での中断点を設定し、そのときの引数あるいはＣＯＭＭＯＮ並びの値を参照し、モジュール間のデバッグ作業を行うことができる。また、この初期表示において、利用者はコールグラフかソースプログラム、あるいはその両方の表示を選択することができる。さらに、コールグラフ上で指定した任意のモジュールのソースプログラムに表示内容を変更することもできる。

【００１０】図３は、ソースファイル１中に格納されているＦＯＲＴＲＡＮソースプログラムの一例であり、このＦＯＲＴＲＡＮソースプログラムは、プログラム実行時にモジュールＳＵＢの実行文１１において０による割算が発生して異常終了するプログラムの例である。

【００１１】図４（ａ）は、コンパイル時に格納されるソース解析情報３を示す図である。すなわち、ソース解析情報３は、モジュール単位のアドレス、名称の使用位置、引用している手続き、引数等の情報で構成されている。また、図中の１２は、配列Ａの先頭アドレスであり、１３は、ＳＵＢの入口のアドレスであり、１４、１５、１６は、それぞれ引数Ｐ，Ｑ，Ｒのアドレスである。

【００１２】図４（ｂ）は、ロードモジュール６を示す。すなわち、プログラム実行時に各変数、配列の値が設定される領域の一部を示し、１７、１８、１９は、それぞれ引数Ｐ，Ｑ，Ｒの値の位置であり、２０は、配列Ａの要素値の位置である。

【００１３】以下、図３のプログラム実行時におけるプ

ログラムデバッガ７を用いたデバッグ方法について詳細に説明する。まず、コンソール８の表示ウィンドウは、図５（ａ）に示すように初期表示される。該表示ウィンドウにおいて、利用者が実行ボタン２１を押下することによって、プログラムが実行される。

【００１４】プログラムデバッガ７は、コールグラフ表示２２上の実行中モジュールの変化に対応してソースプログラム表示２３の表示内容を更新する。そして、図５（ｂ）に示すように、コールグラフ表示２２上で異常終了したモジュールＳＵＢを高輝度表示し、またソースプログラム表示２３上で異常終了した実行文（前述したように、０による割算）の位置２４を高輝度表示して、実行を終了する。

【００１５】次いで、不良が発生しているモジュールが特定されると、図６（ａ）に示すように、利用者がモジュールＳＵＢの入口に中断点２５を設定する。プログラムデバッガ７は、設定された実行中断点２５における引数値を表示する（図６（ａ）の２６）。

【００１６】すなわち、モジュールＳＵＢの入口でプログラムの実行を中断したとき、ロードモジュール６の内容は、図６（ｂ）に示す状態になっている。プログラムデバッガ７は、図４に示す解析情報中の引数のアドレス１４，１５，１６に対応したロードモジュール位置１７，１８，１９から引数値を得て、それらの値を引数名と共に表示する。このように、モジュールＭＡＩＮとモジュールＳＵＢの間で受渡しされる引数とその値が表示されるので、引数Ｐが値０で渡されていることがその表示内容から分かる。

【００１７】このようにして、本実施例ではモジュール間で受渡しされるデータの参照を容易に行うことができ、不良のあるモジュールを特定することができる。上記した実施例では、その表示内容からモジュールＳＵＢの呼び出し元であるＭＡＩＮが不正な値を渡していると判断することができる。

【００１８】また、不良が発生しているモジュールが特定されると、任意のソースプログラム行に実行中断点を設定したり、あるいはソースプログラム上から、変数の値が現状の値と変化した時点で実行を中断する変数を選択したり、任意のソースプログラム行に対して実行中断条件の指定を行うことにより、プログラム実行を中断し、プログラム実行中の任意の箇所での値の参照を行うこともできる。

【００１９】図１は、ＭＡＩＮモジュールの実行文に中断点３１を設定し、プログラムの実行が該中断点３１で中断された状態で、利用者がソースプログラム上の配列Ａの要素の参照をマウスによって指示したときに（行２６のプログラム文中で“Ａ”を指示）、配列Ａの配列要素がコンソール８上に二次元表示される図である。

【００２０】図１において、添字設定部２７で可変とする次元のｘ軸、ｙ軸を設定する。この例では、ｘ軸方向



がＩ、ｙ軸方向がＫとなり、Ｊは値２に固定され、ｘ，ｙがそれぞれ１から５の範囲の配列要素の値が二次元表示部２８に表示されている。また、上下左右のスクロールボタン２９を操作することにより、配列要素の値を自由に二次元表示部２８に位置付けて参照することができ、更に不連続な配列要素の値も容易に参照することができる。

【００２１】この配列要素の二次元表示は次のようにして行われる。図８は、本実施例のプログラムデバッガ７における二次元表示の処理フローチャートである。すなわち、中断点の位置３１でプログラム実行が中断したとき、ロードモジュール６の内容は、図７に示した状態になっている。プログラムデバッガ７は、図８の処理に従って、ユーザによって指定された配列Ａ（ｘ，２，ｙ）の値を二次元表示する。

【００２２】図８において、プログラムデバッガ７は、図４（ａ）の解析情報中の配列Ａの先頭アドレス１２を取り出す（ステップ３２）。配列Ａの要素（１，２，１）、（２，２，１）、（３，２，１）、（４，２，１）、（５，２，１）．．．．のアドレスを、配列の上限まで求める（ステップ３３、３４、３５）。図９は、配列Ａの要素位置を示し、図中３８は、ｘ＝１、ｙ＝１からｘ＝５、ｙ＝１（Ｊは２に固定）までの要素の位置を示す。

【００２３】プログラムデバッガ７は、計算された図９の位置３８～４２のアドレスをキーにして、ロードモジュール６の位置２０を参照して、要素値を次々に読み出す（ステップ３６）。そして、読み出された配列の各要素を図１に示すように二次元表示する（ステップ３７）。

【００２４】この例の場合、配列Ａの要素値の表示において、３０で指示される表示部には不正なデータ（データ値＝０）が表示されている。これは図３の型宣言文９で宣言されている配列Ｂ、Ｃの値が、実行文１０によって配列外参照されて（つまり、Ｂ（Ｉ，Ｊ，５），Ｃ（Ｉ，Ｊ，５））、配列Ａの値が生成されているからである。そして、図９のＫ＝５の配列要素群には、不正なデータが設定されていて、この結果、図１の表示部３０には不正なデータ（データ値＝０）が表示されることになる。

【００２５】このように本実施例によれば、配列を二次元表示することによって、不正なデータの確認を容易に行うことができる。さらに、この不正データの確認時点でプログラムの動作確認を行う場合には、表示部３０に表示されている要素値に正しい値を設定して、プログラムを実行することにより確認することができる。

【００２６】また、配列Ａの値を定義点（図３の１０の位置）で参照する場合には、プログラムを再実行し、図１０の実行文に中断点４４を設定して値を参照する。しかし、この文はＤＯループ中にあるため、そのままでは

5

各ループ毎に実行が中断してしまう。そこで、本実施例のプログラムデバッガ７には、各中断点毎に中断までの繰返し実行回数の設定を行えるようなウィンドウ４３が用意され、ループ中の実行文でも繰返し実行回数の任意の値の時にプログラムを容易に中断することができ、利用者のデバッグ効率の向上を図っている。

【００２７】

【発明の効果】以上、説明したように、本発明によれば、配列要素を個々に指定することなく、配列要素の値を二次元表示画面上で任意に参照することができるので、配列要素中の不正な値の検出を容易に行うことができ、デバッグ作業の効率を大幅に向上させることができる。

【図面の簡単な説明】

【図１】本発明により配列の各要素が二次元表示された図である。

【図２】本発明のデバッグ方法が適用されるデバッガのシステム構成図である。

【図３】ＦＯＲＴＲＡＮソースプログラムの一例である。

【図４】（ａ）は、コンパイル時に格納されるソース解析情報であり、（ｂ）は、ロードモジュールである。

6

【図５】（ａ）はコンソールの初期表示画面であり、（ｂ）は異常終了したモジュールを高輝度表示した画面である。

【図６】（ａ）はモジュールＳＵＢの入口に中断点を設定した表示画面、（ｂ）はロードモジュールの内容の一部である。

【図７】プログラム実行が中断したときのロードモジュールの内容である。

【図８】本実施例のプログラムデバッガにおける二次元表示の処理フローチャートである。

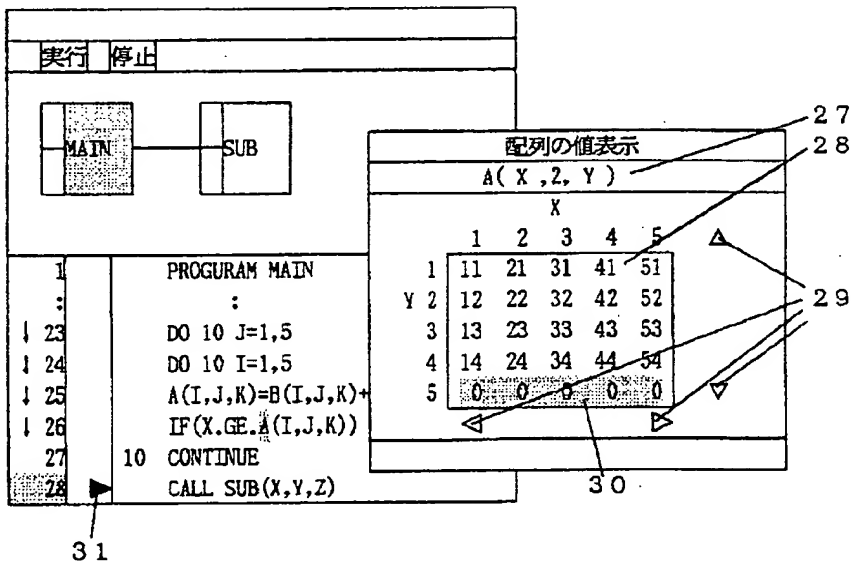
【図９】配列Ａの要素位置を示す図である。

【図１０】デバッガに用意された中断までの繰返し実行回数の設定を行うための表示画面である。

【符号の説明】

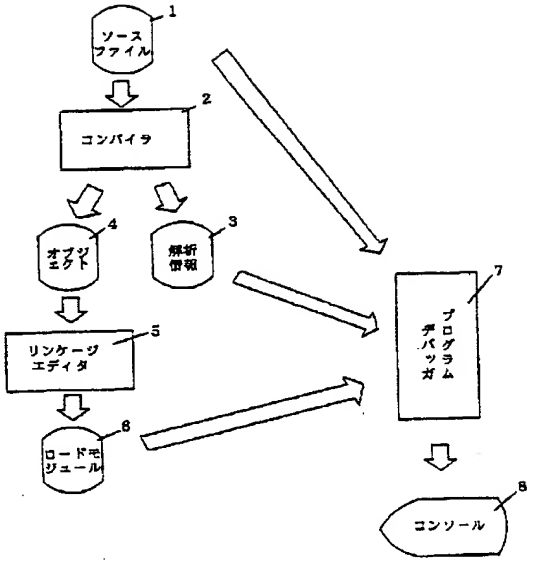
1　ソースファイル
2　コンパイラ
3　解析情報
4　オブジェクトプログラム
5　リンケージエディタ
6　ロードモジュール
7　プログラムデバッガ
8　コンソール

【図１】

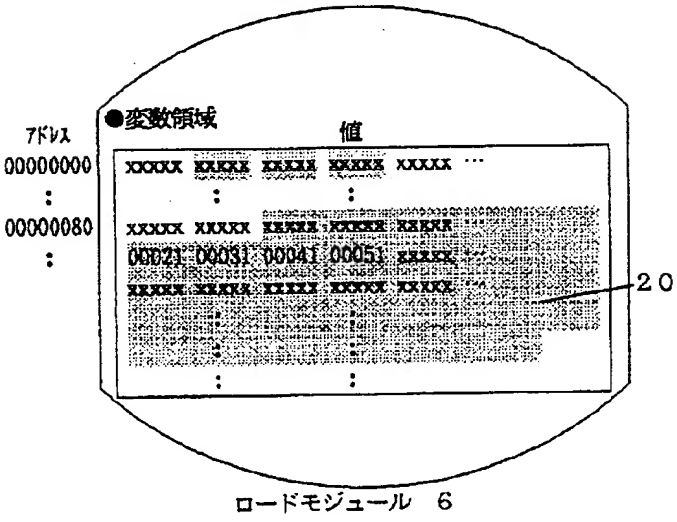


【図２】

システム構成図

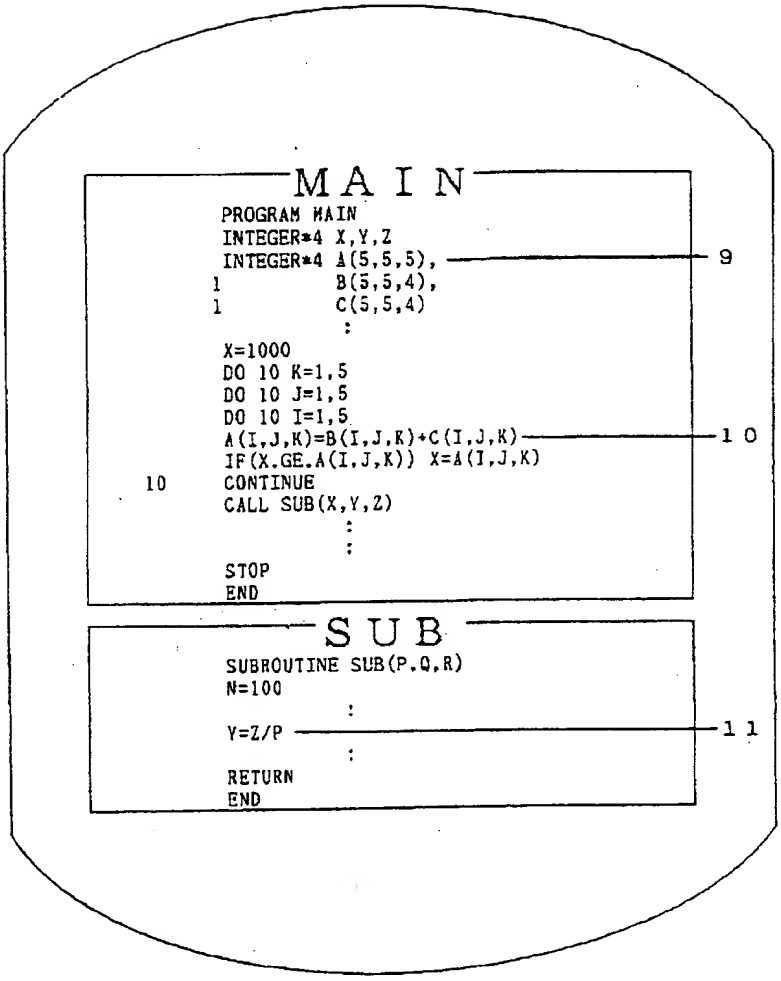



【図７】



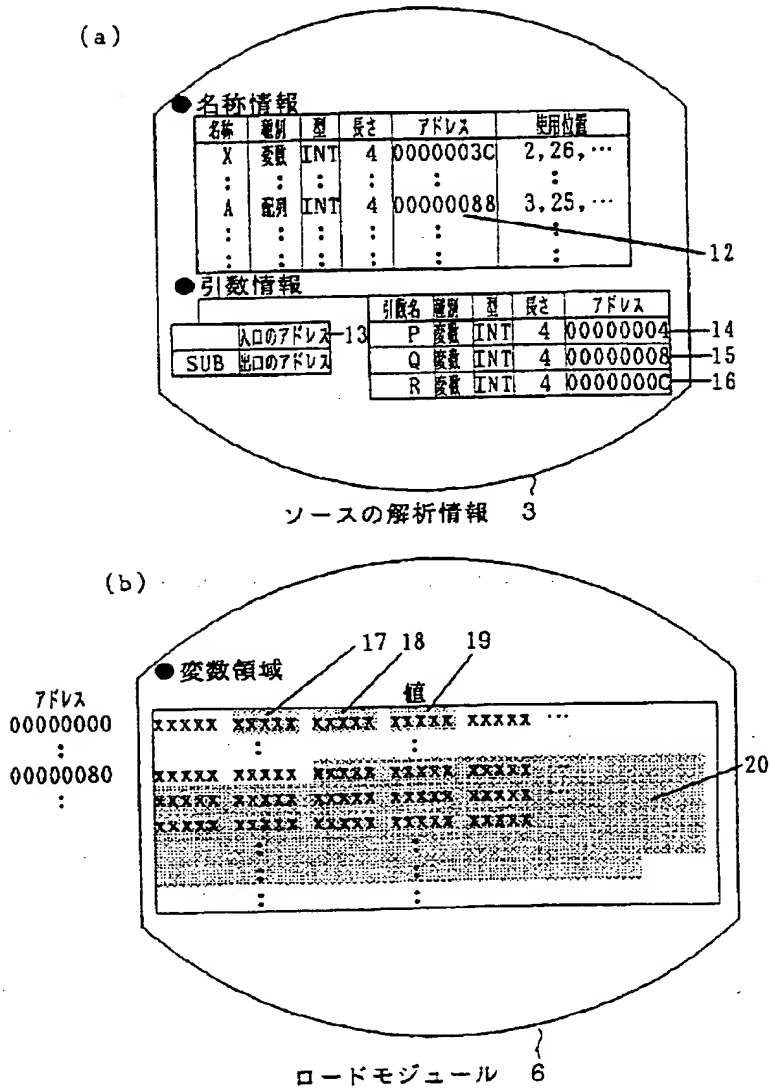
【図3】

ソースプログラム例

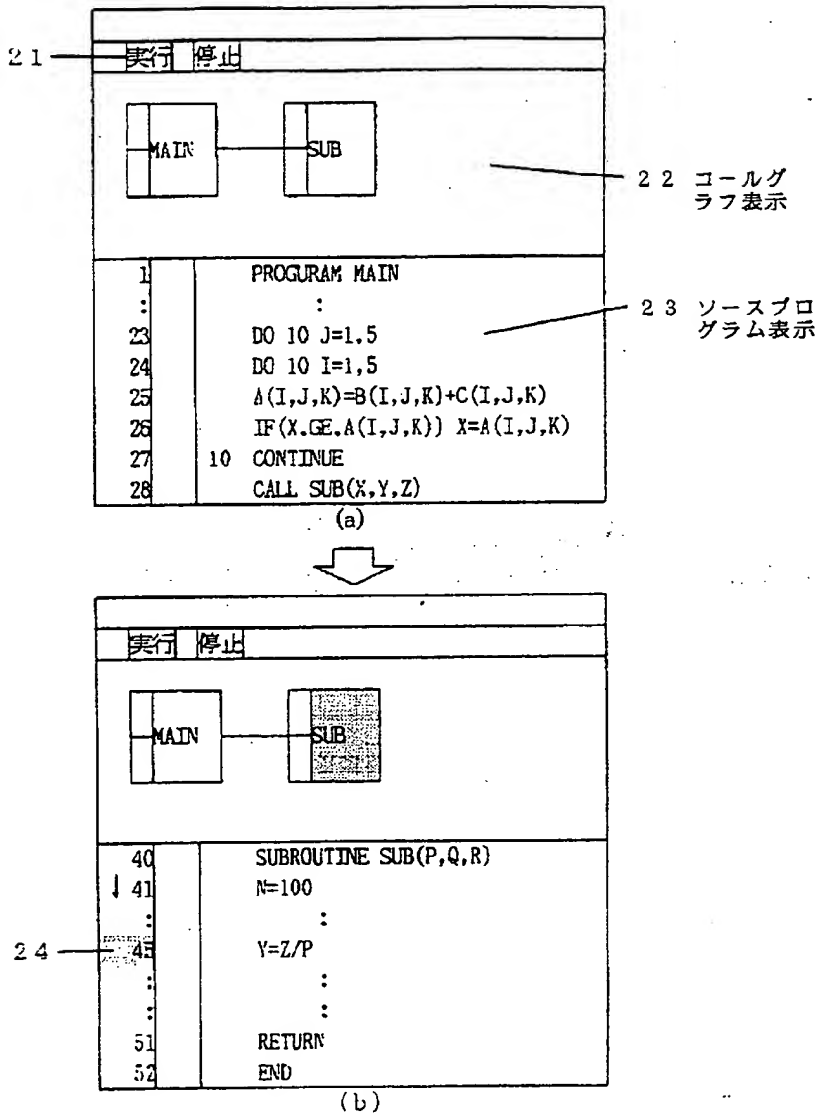


【図4】

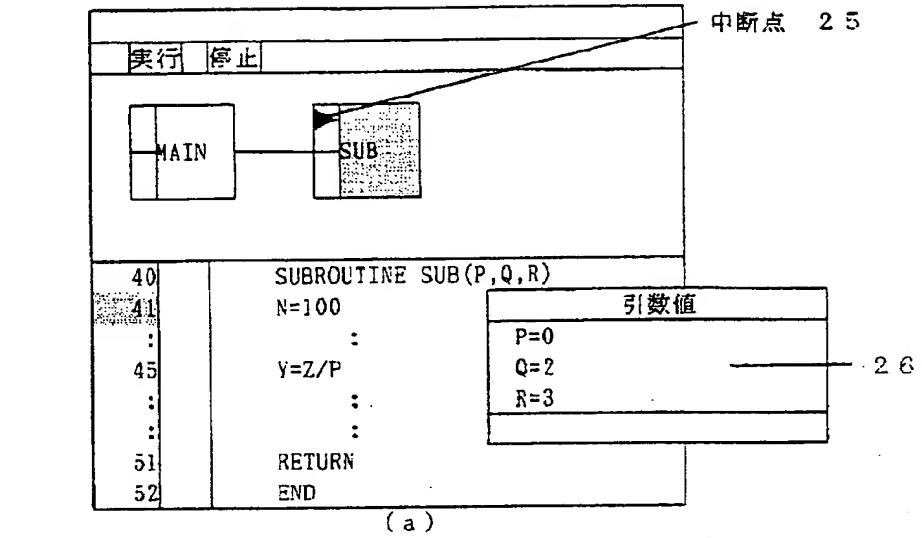
解析情報　とロードファイル



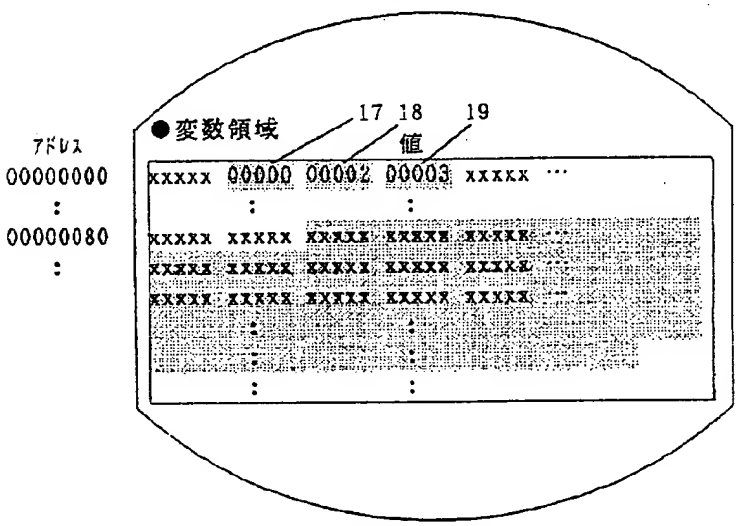
【図5】


21
実行
停止
MAIN
SUB
22 コールグラフ表示
PROGRAM MAIN
23 ソースプログラム表示
DO 10 J=1,5
DO 10 I=1,5
A(I,J,K)=B(I,J,K)+C(I,J,K)
IF(X.GE.A(I,J,K)) X=A(I,J,K)
10 CONTINUE
CALL SUB(X,Y,Z)
(a)
実行
停止
MAIN
SUB
SUBROUTINE SUB(P,Q,R)
N=100
24
Y=Z/P
RETURN
END
(b)

【図6】




(a)

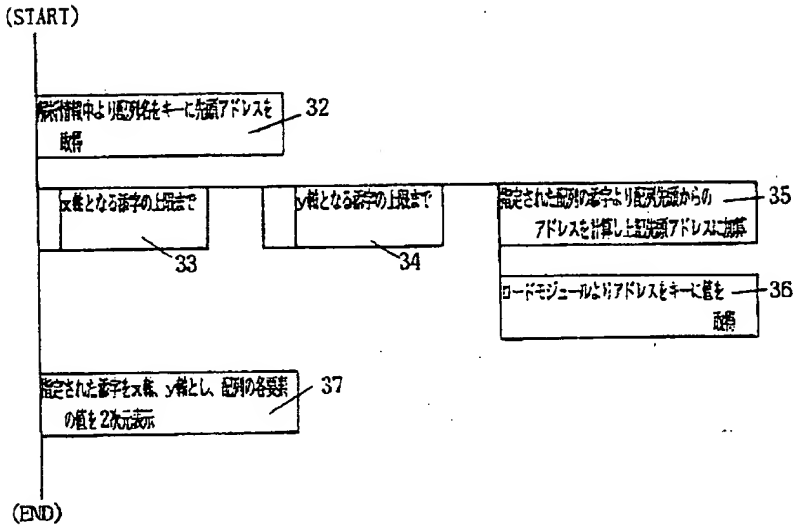



ロードモジュール　6

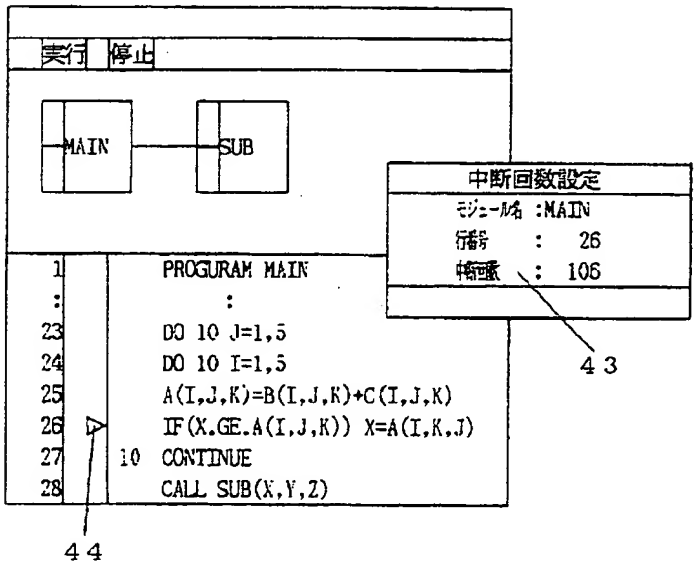
(b)

【図８】

２次元表示処理の流れ



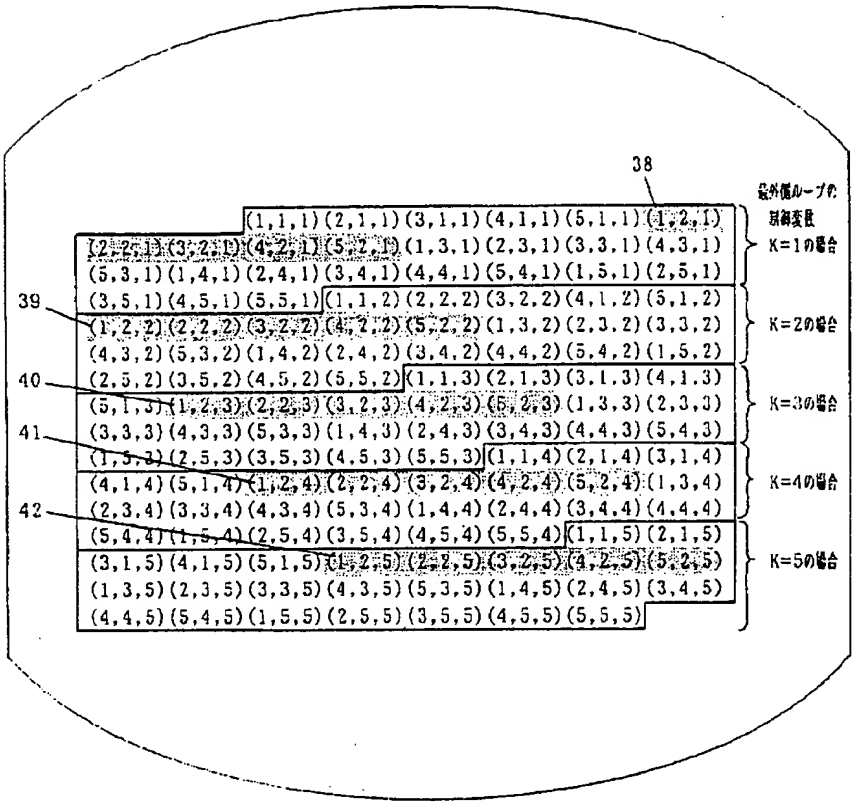

【図１０】



【図9】

配列Aの要素位置

［配列Aの要素のアドレス上の位置］




---

フロントページの続き

(72)発明者　佐藤　真琴
東京都国分寺市東恋ケ窪１丁目280番地
株式会社日立製作所中央研究所内

(72)発明者　山田　重己
神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウェア開発本部内